2017 21st International Conference on Process Control (PC)
June 6–9, 2017, Štrbské Pleso, Slovakia

Novel tools for model-based control system design based on FMI/FMU standard with application in energetics

Martin Čech and Jana Königsmarková and Jan Reitinger and Pavel Balda

NTIS - New Technologies for the Information Society
Faculty of Applied Sciences, University of West Bohemia
Technicka 8, 306 14 Pilsen, Czech Republic
Email: mcech@ntis.zcu.cz, jkonig@ntis.zcu.cz, reitinge@ntis.zcu.cz, pbalda@ntis.zcu.cz

Abstract—**The paper presents novel tools for model-based control system design based on FMI/FMU standard (Functional Mock-up Interface / Unit). It is focused on application of FMI standard for easy integration of control system development cycle starting with Model-in-the-Loop (MIL) simulation and finishing with Hardware-in-the-Loop (HIL) simulation. It is shown, how the Functional Mock-up Units (FMU) containing dynamic differential-algebraic equations of various parts of the device (mechanical, electrical, hydraulic, thermal, etc.) can be easily deployed to unified simulation environment where the control system is designed, consequently. The procedure allows to combine inputs from various Modelica-based tools at the process model side, utilizing power of Matlab/Simulink for design, analysis and optimization of control system and perform final test via HIL scenario where both the model and control system are simulated in real-time on separated HW units. The pros and cons of both FMI concepts, i.e. Co-Simulation and Model Exchange are discussed in detail. The whole procedure is demonstrated on a steam turbine example combining component-based and equation based modeling. Both the turbine model and the full control loop are validated in all phases of control system development. It is shown, that monolithic simulation block with proprietary solver reduces computational burden compared to automatic FMU concept.**

Index Terms—**model-based design, FMI, FMU, Model-in-the-Loop, Software-in-the-Loop, Hardware-in-the-Loop, steam turbine control**

I. Introduction

There are various primary aims of creating models, such as product design or decision making support. In this paper we follow the problem of cyber-physical system design where the control part always plays an unsubstituable role. In the past, the control part was usually designed for already existing machine or process technology. Such approach led only to sub-optimal control quality. Nowadays, with increasing computational power of computers and capabilities of modeling/simulation platforms, the control system can be designed and optimized simultaneously with machine/process development. Such procedure usually passes through well known phases Model-in-the-Loop, Software/Processor-in-the-Loop and Hardware-in-the-Loop. Seamless passing through those phases generates strict requirements on the modeling and simulation platforms, such as *multi-domain* and *multi-level* capability [1]–[3], *interaction* with real world (see [3]–[5]) and, consequently, the ability of deterministic execution and *real-time* simulation (see e.g. [6], [7]). The following 'state of the art' challenges has been identified:

Firstly, the complexity of systems requires to combine models developed via various tools and methods, both equation based [8], [9] and component based [10]–[12]. At present, the FMI *(Functional Mock-up Interface)* standard is a powerful tool to cope with this problem [13]–[16]. It allows to export models from various platforms (primarily based on Modelica language) as FMUs *(Functional Mock-up Units)* and integrate them with target simulation platform where a control system is usually designed (e.g. Matlab/Simulink™ [17], NI Labview™). However, real-time execution of FMUs is still a problem which has not been solved satisfactorily yet.

Next, the features of the numerical solver (for system of ODE - *Ordinary differential equations*, DAE - *Differential algebraic equations*) must be fully under control with ability to set the number of iterations, maximal error, etc. in order to execute each step within the available time-frame of fixed sampling period. For time critical simulations, it is sometimes necessary to develop own optimized solver code, see. e.g. [18] where the fast solver implementation on FPGA is described.

Finally, the existing commercial products and related libraries (e.g. Dymola™) are extremely expensive and the resulting models cannot be fully customized.

Consequently, the main aim of the paper is to present the set of novel tools for model-based control system design which help to mitigate above mentioned drawbacks and validate them on a steam turbine model (derived in [19], [20]). Although some fuzzy logic approaches to turbine modeling has been investigated [21], here a DAE (differential algebraic equation) model described e.g. in [19], [22] has been adopted. The set includes specific solvers, tools for FMU import onto real-time platform and tools for HIL simulation support. Following the approach from [10]–[12], the models are firstly developed on Modelica platform, deployed to Matlab/Simulink and compared to block-based and equation based Simulink implementation. Further the model is validated also at real-time platform (Stage I). As a next step, the control system is designed here following again the full MIL-SIL/PIL-HIL approach (Stage II).

The paper is organized as follows: In Section II, the general problem of seamless control system development is stated and the gaps of state of the art solutions highlighted. Section III summarizes the main technological contribution of the paper, i.e. SW and HW modules creating new toolchain for control system development. The whole procedure is validated in Section IV on a complex model of energetic device (controlled steam turbine with shaft and generator). The conclusions and ideas for future work are given in Section V.

II. Problem Formulation - Model Based Control System Design

A. General requirements for complex system modeling

The generic state of the art way of control system design is charted in Fig. 1. Naturally, it is not necessary to pass always through all depicted phases[1]. Anyway, the described approach allows control system co-design in cases when the physical machine/plant does not exist yet, i.e. can be optimized for better controlability during the cycle presented. The procedure generates strict requirements on a modeling and simulation platforms, such as:

- *Multi-domain capability:* The ability to simulate complex system consisting of subsystems from various domains (mechanical, electrical, hydraulic, power, thermal, etc.)
- *Multi-level capability:* The ability to simulate systems at different levels of detail (typically reduced-order models), switch automatically between those levels.
- *Reality in the loop, component in the loop, human in the loop:* The ability to integrate model with existing infrastructure, human behavior or product sub-components. Naturally, this implies the need for real-time simulation ability.
- *Validation and verification:* Reliable tools for model verification and validation. Continuous time model validity check.

B. Handling requirements during control system development

The toolchain developed and presented in this paper supports following steps of control system design:

1) Model creation: The model (in general non-linear) can be obtained in two ways, which can be in principle combined:

- *Semiautomatic deriving of equation* based on known mathematical-physical laws
- *Component based modeling* where the model is created either by custom made or existing libraries based on Modelica language (one can choose between OpenModelica - OM, Dymola, SimulationX or MapleSim).

Point out that the first approach allows future equations optimization and even integration with hand coded solver (often

[1]e.g. in the case when the control structure is simple or the plant/machine is not designed from the scratch

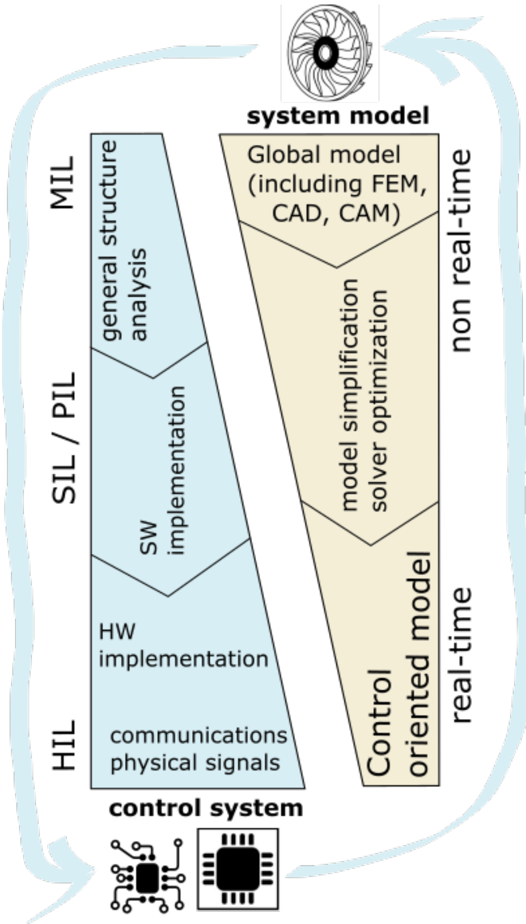

Fig. 1: Generic approach to rapid control system design and development. The system model is parameterized either from primary CAD/CAM sources or via additional identification experiments.

necessary for HIL phase) while the second approach relies on exporting component based model as FMU package (containing both equations and solver[2]). The case study presented in Section IV shows how these principles can be combined using the tools developed.

2) MIL: In this phase the model developed is usually integrated into control system initial design platform (e.g. Matlab/Simulink - used in our case study; NI Labview, etc.) where one uses strong analytical and optimization tools. Point out that Modelica based simulation platforms do not offer necessary libraries for control system design and implementation. In this phase one can test the suitable sampling period, solver type and, if necessary, simplify the model. As shown in Section IV, the developed model can be imported as FMU.

3) SIL-PIL: In this phase, the model is tested with the control system in its implementation environment, either just SW (SIL) or both SW+HW (PIL). The simulation runs in real-time with the optimized solver. One can effectively test

[2]holds true for FMI Co-simulation mode

the control system performance with a selected sampling period. The communication between model and control system is ensured via internal links (i.e. is running on a single computational platform) or via fast industrial communication (EtherCAT, Modbus).

4) HIL: HIL is the final phase before connecting control system to real plant. In this phase the control system is executed on target final HW and communicates with a model simulated in real-time via physical AI/AO, DI/DO signals which can have additional substantial influence on the available bandwidth and even sampling period.

III. Developed tools supporting automatic MIL-SIL-HIL chain

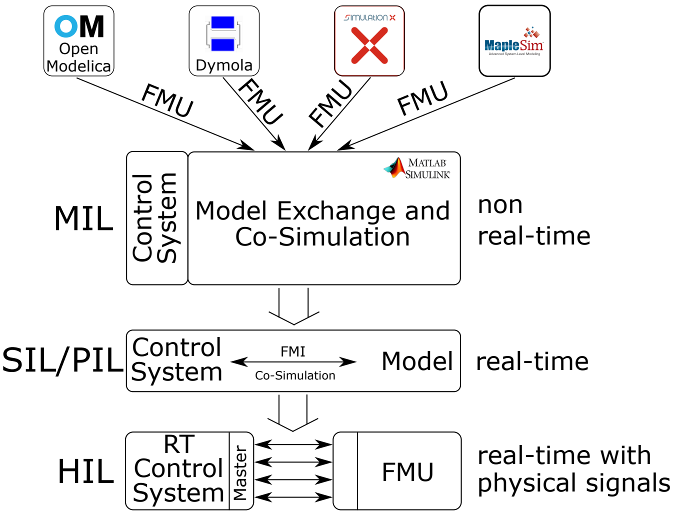

Fig. 2: Using FMI/FMU technology in control system design; the developed tools do support such chain

The set of developed tools supports the toolchain shown in Fig. 2 which precises the generic approach outlined in Fig. 1.

A. SW tools

To support the flowchart mentioned in Fig. 2, the following modules has been developed:

- *FMUCS function block* which allows to import models with FMI 2.0 interface (Co-Simulation mode) into real-time environment, see Fig. 3, full details in [23]. The simulator is deployable to any platform which allows: execute C/C++ code; run user tasks with selected period; has a toolchain for exporting and compiling model into .fmu file; offers suitable communication and data exchange with a control system.
- *Methodology for creating monolithic simulation blocks – MSB* which fits and optimizes the set of differential equations with selected numerical method, both explicit and implicit (e.g. Runge-Kutta, Backward differential formula, RADAUIIA [24]), see Fig. 3.

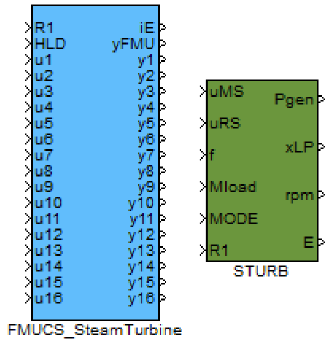

Fig. 3: Developed function block FMUCS allowing to import FMU modules for SIL/PIL and HIL phase; monolithic simulation block STURB; both allow to execute model on proprietary real-time platform

B. HW tools

To support final HIL implementation, a Raspberry Pi containing developed real-time scheduler with lightweight Monarco HAT for data exchange have been adopted.

IV. Case study - steam turbine control system

A. Overall methodology

The overall methodology adopted in a steam turbine use case can be divided into two stages (Fig. 4):

- **Stage I:** The turbine model (in both FMU and MSB form) is passed through all MIL-HIL phases and compared to reference simulation, especially it is tested whether switching from variable to fixed time will not cause loss of precision above an acceptable limit.
- **Stage II:** Starting again at MIL phase, the model (FMU, MSB) is connected to the control system prototype and a procedure depicted in Fig. 2 is executed as described in upcoming subsections.

B. Stage I: Model validation

As the reference solution, the *equation based model* was used – simulated as a total system of ordinary differential equations solved by continuous solver with high precision.

1) MIL phase: We compared the reference data with the FMU exported from OpenModelica (Co-simulation mode, dassl solver, see Fig. 5) and MSB approach (RADAUIIA method). The comparison was done in Matlab using Pilot Support Package for FMU import.

2) SIL-PIL phase: Both FMUCS and STURB blocks were tested in real-time environment on PC.

3) HIL: Both FMUCS and STURB blocks were tested in real-time environment on target model device (Raspberry Pi).

The summary of Stage I can be reviewed in Fig. 9, Tab II and Tab. I for the case of load plugging and unplugging in the standalone mode.

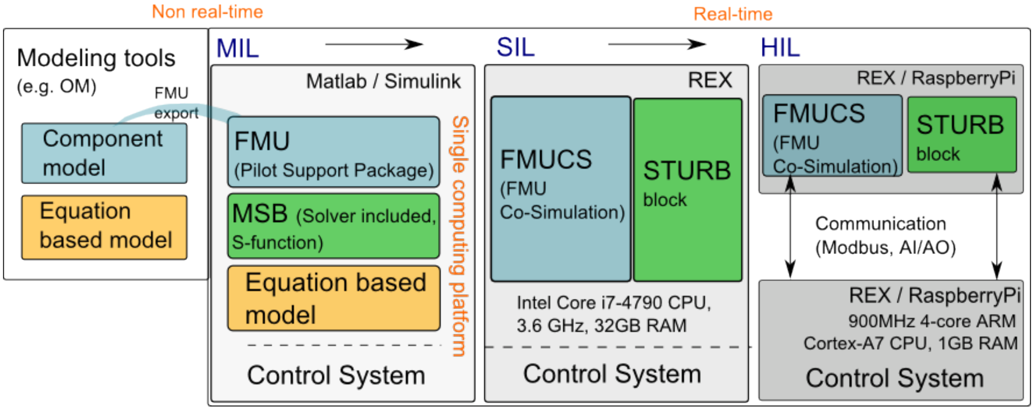

Fig. 4: Steam turbine case study model and control system validation in MIL, SIL, HIL phases (including basic HW parameters): equations derived (orange - reference model), component-based model exported as FMU (blue), monolithic simulation block including proprietary solver (green)

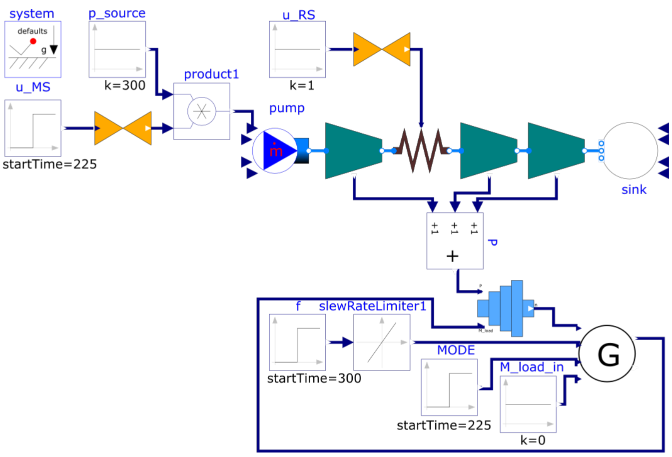

Fig. 5: Component based model of steam turbine which is further exported as FMU to enter the MIL-SIL/PIL-HIL cycle of control design

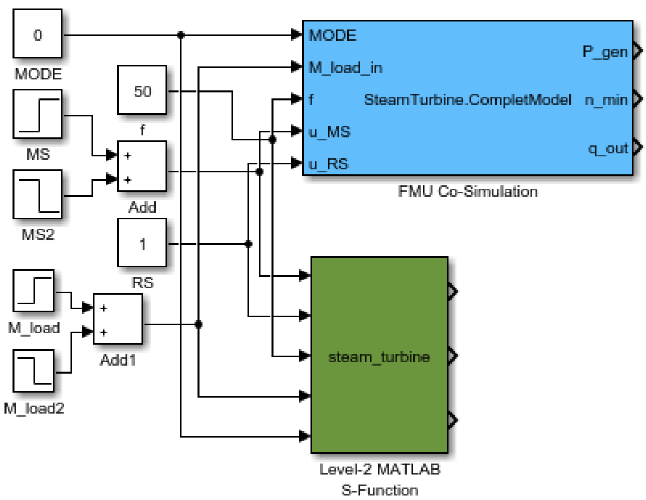

Fig. 6: Model validation schema for Stage I / MIL: FMU exported from OpenModelica environment; MSB approach - S-function

TABLE I: Comparison of maximal relative error of different model implementations; sampling time 5ms.

block	max. relative error
FMU/MIL	0.00060
MSB/MIL	0.00058
FMU/SIL	0.00519
MSB/SIL	0.00102

TABLE II: Stage I: Comparison of computational performance of turbine model in real-time environment (SIL - PC+Win10/64, HIL - Raspberry Pi+Raspbian).

block	max time [ms]	average time [ms]
SIL / FMUCS	0.35	0.061
SIL / STURB	0.24	0.025
HIL / FMUCS	0.48	0.11
HIL / STURB	0.34	0.06

C. Stage II: Controlled model validation

In Stage II, all model validation phases described above were completed by control system. The validation was done for the case of controlling turbine revolutions (rpm). The summary of Stage II can be reviewed in Fig. 10 and Tab III for the scenario when the turbine starts from standstill state, after reaching the nominal network rpms, the load is connected to the turbine.

V. Conclusion

Paper conclusion

In this paper, novel tools for model-based control system design based on FMI/FMU standard have been presented. It was shown how the Functional Mock-up Units (FMUs) having either component-based or equation based origin can

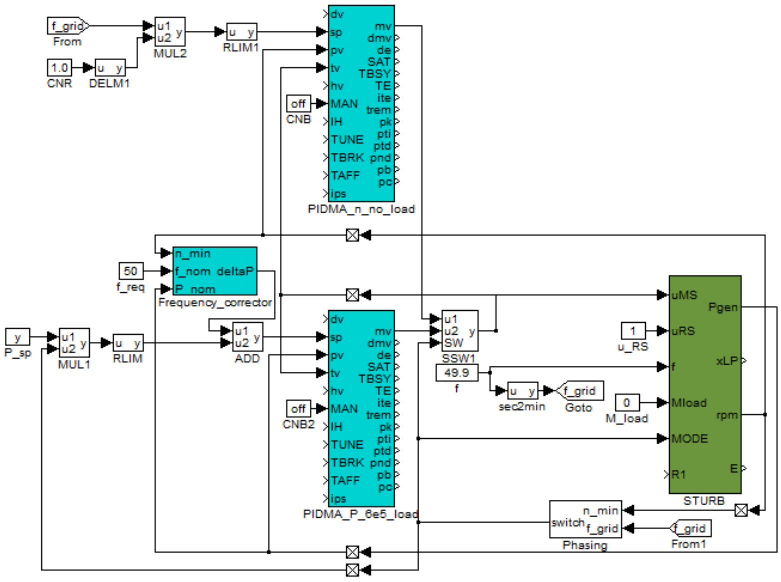

Fig. 7: Controlled model validation schema for Stage II / SIL: Monolithic simulation block STURB is used, simulation running in real-time mode, sampling time 5ms.

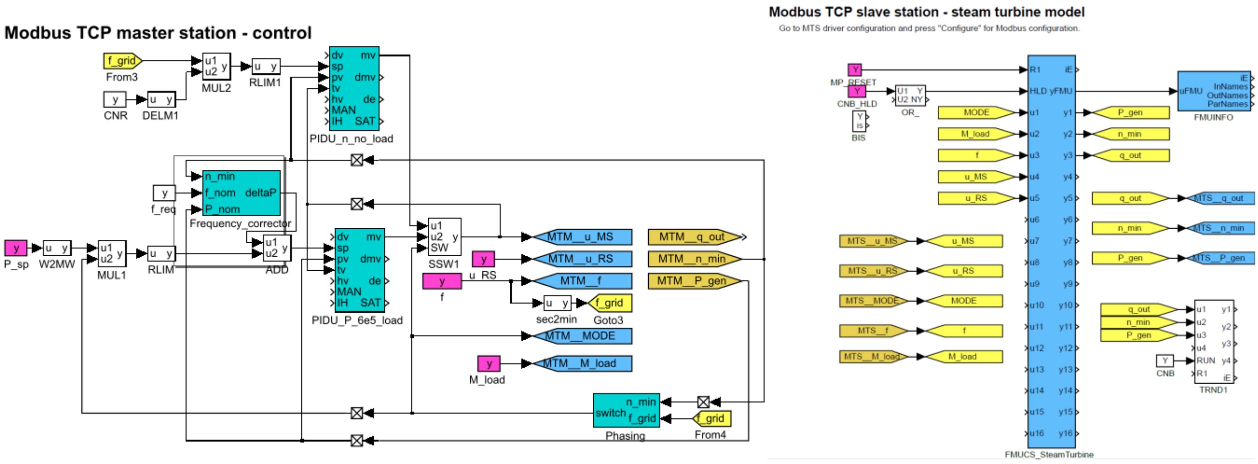

Fig. 8: Turbine control system passing through MIL-SIL/PIL-HIL phases in Stage II; two PID controllers and frequency corrector for different turbine modes, i.e. turbine start up from and steady working point tracking electr. net frequency

TABLE III: Stage II: Comparison of computational performance full simulation (model+control system) in real-time environments SIL - PC+Win10/64 and HIL - Raspberry Pi+Raspbian.

simulation	max time [ms]	average time [ms]
SIL / FMUCS	0.35	0.061
SIL / STURB	0.15	0.014
HIL / control part	0.63	0.07

be deployed from various modeling platforms to unified environment where the control system is designed and optimized, consequently. The whole MIL-SIL-HIL chain described allows in each stage to optimize equations and assign an appropriate solver (even custom developed) fulfilling real-time requirements in final SIL/HIL stages. The whole procedure was demonstrated on an energetic device example – steam turbine with shaft and generator. Both model and full control loop simulation were validated and compared in all stages. It was shown, that monolithic simulation block with proprietary solver reduces computational burden compared to automatic FMU concept.

Ideas for future work

The future research covers several topics: deep knowledge, analysis and optimization of solver which are assigned/deployed via automatic procedures. Full support of FMI/Model Exchange mode, i.e. full integration of custom

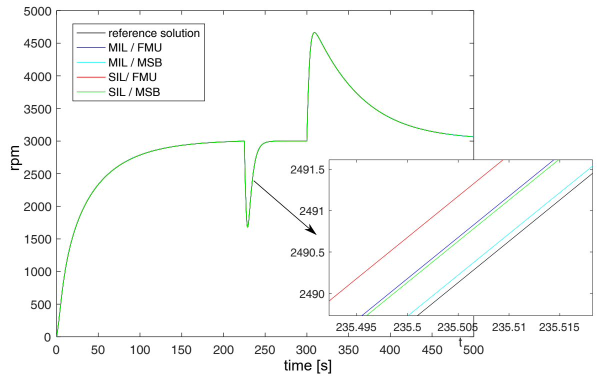

Fig. 9: Model simulation passing Stage I, for inputs: $u_{RS} = 1$, $MODE = 0$, $f = 50$, $u_{MS} = 0.0333$ for $t \in \langle 0, 225 \rangle \cup (300, 500)$ and $u_{MS} = 0.6613$ for $t \in (225, 300\rangle$, $M_{load} = 0$ for $t \in \langle 0, 225 \rangle \cup (300, 500)$ and $M_{load} = 6e5$ for $t \in (225, 300\rangle$.

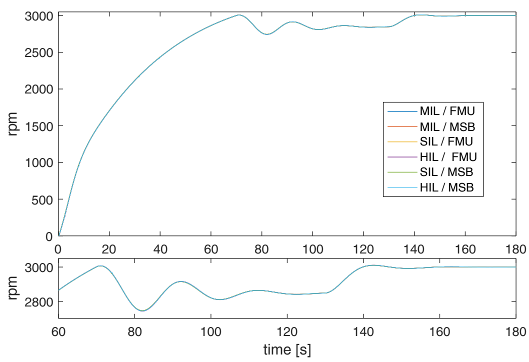

Fig. 10: Controlled model passing Stage II, in $t = 70$, the load is connected to turbine

build solver into SIL/PIL and HIL stage. Further the aim is to develop scalable set of models of selected energetic devices which can be connected into simulated network and consequently an appropriate higher level control system is designed. The models will be deployed to embedded HIL simulators (e.g. Raspberry Pi based). The authors believe that such models will be useful for all academic, trainee, and industrial purposes.

Acknowledgement

This work was supported by the Technology Agency of the Czech Republic – project No. TA04010364 and by the project LO1506 of the Czech Ministry of Education, Youth and Sports under the program NPU I. The support is gratefully acknowledged.

References

[1] F. Casella and A. Ranade, "Efficient modelling and simulation of multi-domain smart grids using modelica and multi-rate integration algorithms," 2016, pp. 6285–6291.

[2] B. Yang, X. Wu, and W. Li, "FMI based multi-domain modeling and simulation for aircraft power distribution system," 2016, pp. 325–330.

[3] C. Molitor, A. Benigni, A. Helmedag, K. Chen, D. Cali, P. Jahangiri, D. Müller, and A. Monti, "Multiphysics test bed for renewable energy systems in smart homes," *IEEE Transactions on Industrial Electronics*, vol. 60, no. 3, pp. 1235–1248, 2013.

[4] T. Hong, H. Sun, Y. Chen, S. Taylor-Lange, and D. Yan, "An occupant behavior modeling tool for co-simulation," *Energy and Buildings*, vol. 117, pp. 272–281, 2016.

[5] S. Gaaloul, H.-A. Dang, A. Kashif, B. Delinchant, and F. Wurtz, "A new co-simulation architecture for mixing dynamic building simulation and agent oriented approach for users behaviour modelling," 2013, pp. 3225–3233.

[6] D. Broman, C. Brooks, L. Greenberg, E. Lee, M. Masin, S. Tripakis, and M. Wetter, "Determinate composition of fmus for co-simulation," 2013.

[7] K. Jha, S. Mishra, and A. Joshi, "Boost-amplifier-based power-hardware-in-the-loop simulator," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 12, pp. 7479–7488, 2015.

[8] J.-H. Jeon, J.-Y. Kim, H.-M. Kim, S.-K. Kim, C. Cho, J.-M. Kim, J.-B. Ahn, and K.-Y. Nam, "Development of hardware in-the-loop simulation system for testing operation and control functions of microgrid," *IEEE Transactions on Power Electronics*, vol. 25, no. 12, pp. 2919–2929, 2010.

[9] J. Wang, Y. Song, W. Li, J. Guo, and A. Monti, "Development of a universal platform for hardware in-the-loop testing of microgrids," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 4, pp. 2154–2165, 2014.

[10] D. Perera, D. Winkler, and N.-O. Skeie, "Multi-floor building heating models in MATLAB and Modelica environments," *Applied Energy*, vol. 171, pp. 46–57, 2016.

[11] C. Martin-Villalba, A. Urquia, and S. Dormido, "Development of an industrial boiler virtual-lab for control education using Modelica," *Computer Applications in Engineering Education*, vol. 21, no. 1, pp. 36–45, 2013.

[12] J. Baladi, N. Khan, M. Rabbani, K. M.S., and S. Asghar, "Modeling of condensate water system in a nuclear power plant with new control strategy using Dymola/Modelica," *International Journal of Innovation and Applied Studies*, 2014.

[13] MODELISAR, "Functional mock-up interface for model exchange and co-simulation, version 1.0," 2010.

[14] Modelica Association Project FMI, "Functional mock-up interface for model exchange and co-simulation, version 2.0," 2014.

[15] E. Tsampasis, L. Sarakis, H. Leligou, T. Zaharidis, and J. Gerofalakis, "Novel simulation approaches for smart grids," *Journal of Sensor and Actuator Networks*, 2016.

[16] T. Noidui and M. Wetter, "Tool coupling for the design and operation of buildingenergy and control systems based on the functional mock-up interface standard," 2014, pages=311-320,.

[17] E. Atlas, M. Erdogan, O. Ertin, A. Guclu, Y. Saygi, U. Kaynak, and C. Kasnakoglu, "Hardware-in-the-loop test platform design for uav applications," *Applied Mechanics and Materials*.

[18] A. Hasanzadeh, C. Edrington, N. Stroupe, and T. Bevis, "Real-time emulation of a high-speed microturbine permanent-magnet synchronous generator using multiplatform hardware-in-the-loop realization," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 6, pp. 3109–3118, 2014.

[19] G. Zimmer, "Modelling and simulation of steam turbine processes: Individual models for individual tasks," *Mathematical and Computer Modelling of Dynamical Systems*, vol. 14, no. 6, pp. 469–493, 2008.

[20] J. Reitinger, P. Balda, and M. Schlegel, "Steam turbine hardware in the loop simulation," *Prepring submitted to 21st International Conference on Process Control*, 2017.

[21] P. Sokolski, T. A. Rutkowski, and K. Duzinkiewicz, "Simplified, multi-regional fuzzy model of a nuclear power plant steam turbine," in *2016 21st International Conference on Methods and Models in Automation and Robotics (MMAR)*. IEEE, 2016, pp. 379–384.

[22] A. Chaibakhsh, AliGhaffari, "Steam turbine model," *Simulation Modelling Practice and Theory*, vol. 16, no. 9, pp. 1145–1162, 2008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1569190X08001196

[23] P. Balda, "Real-time simulator of component models based on functional mock-up interface 2.0," *Prepring submitted to 21st International Conference on Process Control*, 2017.

[24] E. Hairer and G. Wanner, "Solving ordinary differential equations. ii, volume 14 of springer series in computational mathematics," 1996.